

Flight Gate Assignment with a Quantum Annealer

Tobias Stollenwerk, Elisabeth Lobe, Martin Jung

German Aerospace Center (DLR)
SIAM Annual Meeting, Portland

A large, curved image of the Earth from space occupies the bottom right portion of the slide. It shows a view of the Earth's horizon with blue oceans, green landmasses, and white clouds. The text "Knowledge for Tomorrow" is overlaid on this image in a white, sans-serif font.

Knowledge for Tomorrow

Flight Gate Assignment

A day at Frankfurt Airport

- about 1300 aircraft movements (arrival and departure)
- more than 90% are passenger flights
- more than 170000 passengers
- about 60% transfer passengers
- 278 gates



Passenger Flows

- F, G sets of flights and gates

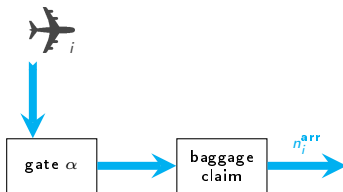


gate α

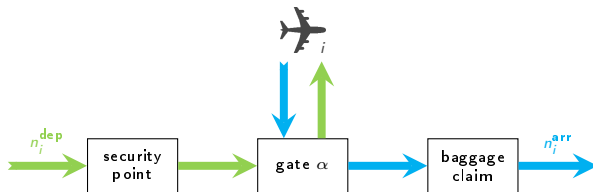


Passenger Flows

- F, G sets of flights and gates



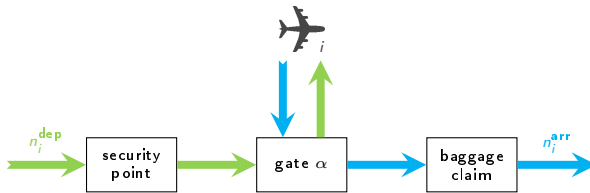
Passenger Flows



- F, G sets of flights and gates
- $n_i^{\text{dep/arr}}$ passengers which depart/ arrive with flight i



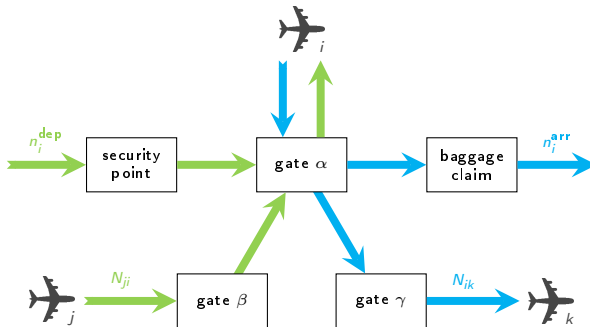
Passenger Flows



- F, G sets of flights and gates
- $n_i^{\text{dep/arr}}$ passengers which depart/ arrive with flight i



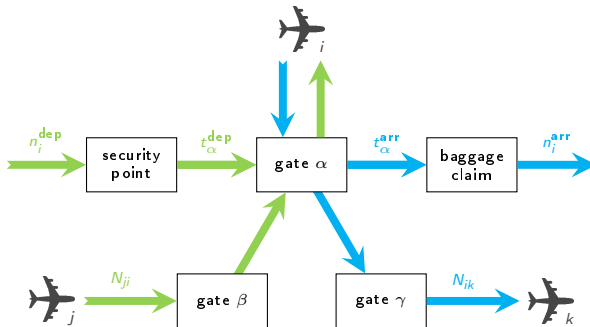
Passenger Flows



- F, G sets of flights and gates
- $n_i^{\text{dep/arr}}$ passengers which depart/ arrive with flight i
- N_{ij} transfer passengers from flight i to j



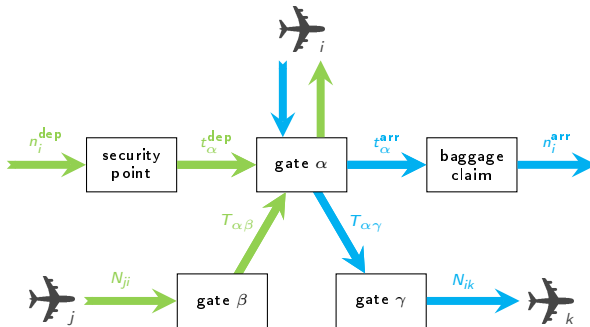
Passenger Flows



- F, G sets of flights and gates
- $n_i^{\text{dep/arr}}$ passengers which depart/ arrive with flight i
- N_{ij} transfer passengers from flight i to j
- $t_\alpha^{\text{dep/arr}}$ average time to arrive at/ leave from gate α



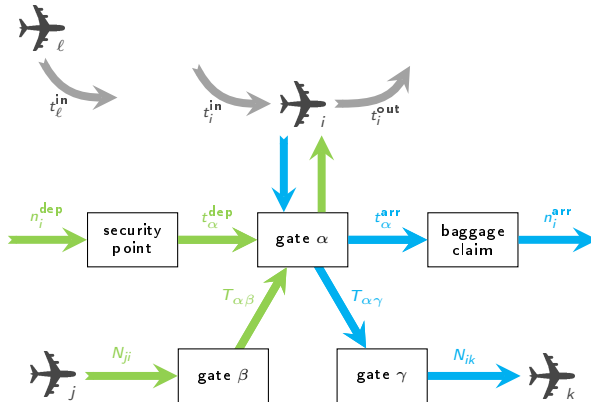
Passenger Flows



- F, G sets of flights and gates
- $n_i^{dep/arr}$ passengers which depart/ arrive with flight i
- N_{ij} transfer passengers from flight i to j
- $t_{\alpha}^{dep/arr}$ average time to arrive at/ leave from gate α
- $T_{\alpha\beta}$ average time to get from gate α to β



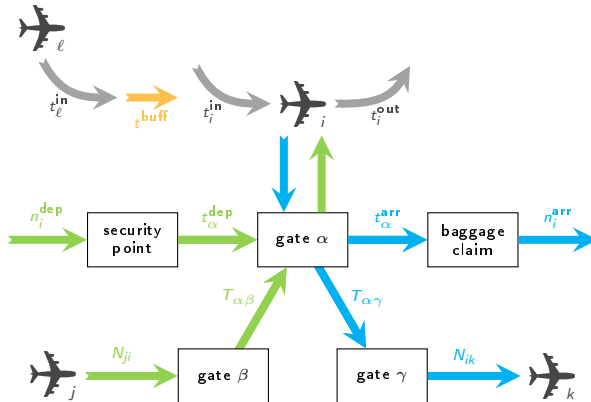
Passenger Flows



- F, G sets of flights and gates
- $n_i^{\text{dep/arr}}$ passengers which depart/ arrive with flight i
- N_{ij} transfer passengers from flight i to j
- $t_\alpha^{\text{dep/arr}}$ average time to arrive at/ leave from gate α
- $T_{\alpha\beta}$ average time to get from gate α to β
- $t_i^{\text{in/out}}$ arrival/departure time of flight i



Passenger Flows

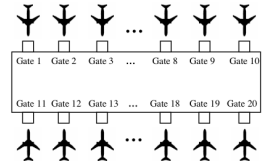


- F, G sets of flights and gates
- $n_i^{\text{dep/arr}}$ passengers which depart/ arrive with flight i
- N_{ij} transfer passengers from flight i to j
- $t_\alpha^{\text{dep/arr}}$ average time to arrive at/ leave from gate α
- $T_{\alpha\beta}$ average time to get from gate α to β
- $t_i^{\text{in/out}}$ arrival/departure time of flight i
- t_{buff} buffer time between two flights at the same gate



Question

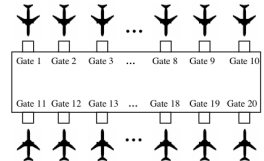
Which flight should be assigned to which gate, such that the total transit time of the passengers in minimal?



Question

Which flight should be assigned to which gate, such that the total transit time of the passengers is minimal?

$$A : F \rightarrow G$$

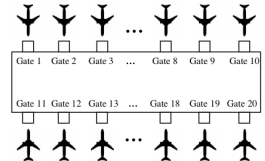


Question

Which flight should be assigned to which gate, such that the total transit time of the passengers is minimal?

$$A : F \rightarrow G$$

⇒ Quadratic Assignment Problem



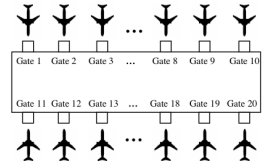
Question

Which flight should be assigned to which gate, such that the total transit time of the passengers is minimal?

$$A : F \rightarrow G$$

⇒ Quadratic Assignment Problem

- fundamental problem in combinatorial optimization



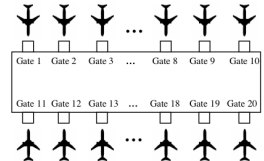
Question

Which flight should be assigned to which gate, such that the total transit time of the passengers is minimal?

$$A : F \rightarrow G$$

⇒ Quadratic Assignment Problem

- fundamental problem in combinatorial optimization
- standard formulation is NP-hard



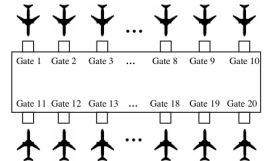
Question

Which flight should be assigned to which gate, such that the total transit time of the passengers is minimal?

$$A : F \rightarrow G$$

⇒ Quadratic Assignment Problem

- fundamental problem in combinatorial optimization
- standard formulation is NP-hard
- seems to exploit possible advantages of the D-Wave machine



FGA Binary Program



α

Variables $x \in \{0, 1\}^{F \times G}$ with

$$x_{i\alpha} = \begin{cases} 1, & \text{if flight } i \text{ takes gate } \alpha, \\ 0, & \text{otherwise} \end{cases}$$



β

γ



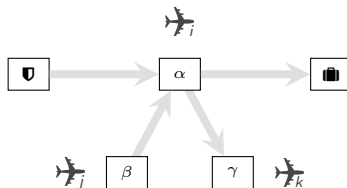
FGA Binary Program

Variables $x \in \{0, 1\}^{F \times G}$ with

$$x_{i\alpha} = \begin{cases} 1, & \text{if flight } i \text{ takes gate } \alpha, \\ 0, & \text{otherwise} \end{cases}$$

Minimizing the total transfer time with objective function

$$O(x)$$

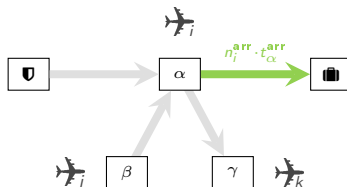


FGA Binary Program

Variables $x \in \{0, 1\}^{F \times G}$ with

$$x_{i\alpha} = \begin{cases} 1, & \text{if flight } i \text{ takes gate } \alpha, \\ 0, & \text{otherwise} \end{cases}$$

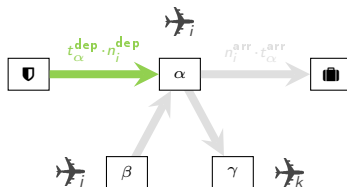
Minimizing the total transfer time with objective function



$$\begin{aligned} O(x) &= O_{\text{arr}}(x) \\ &= \sum_{i\alpha} n_i^{\text{arr}} t_{\alpha}^{\text{arr}} x_{i\alpha} \end{aligned}$$



FGA Binary Program



Variables $x \in \{0, 1\}^{F \times G}$ with

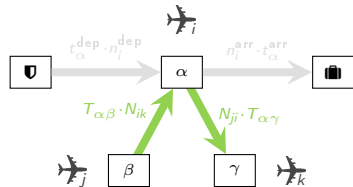
$$x_{i\alpha} = \begin{cases} 1, & \text{if flight } i \text{ takes gate } \alpha, \\ 0, & \text{otherwise} \end{cases}$$

Minimizing the total transfer time with objective function

$$\begin{aligned} O(x) &= O_{\text{arr}}(x) + O_{\text{dep}}(x) \\ &= \sum_{i\alpha} n_i^{\text{arr}} t_{\alpha}^{\text{arr}} x_{i\alpha} + \sum_{i\alpha} n_i^{\text{dep}} t_{\alpha}^{\text{dep}} x_{i\alpha} \end{aligned}$$



FGA Binary Program



Variables $x \in \{0, 1\}^{F \times G}$ with

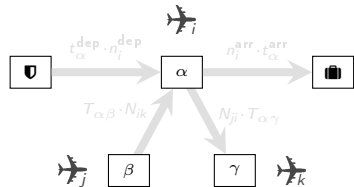
$$x_{i\alpha} = \begin{cases} 1, & \text{if flight } i \text{ takes gate } \alpha, \\ 0, & \text{otherwise} \end{cases}$$

Minimizing the total transfer time with objective function

$$\begin{aligned} O(x) &= O_{\text{arr}}(x) + O_{\text{dep}}(x) + O_{\text{transfer}}(x) \\ &= \sum_{i\alpha} n_i^{\text{arr}} t_{\alpha}^{\text{arr}} x_{i\alpha} + \sum_{i\alpha} n_i^{\text{dep}} t_{\alpha}^{\text{dep}} x_{i\alpha} + \sum_{ij\alpha\beta} N_{ij} T_{\alpha\beta} x_{i\alpha} x_{j\beta} \end{aligned}$$



FGA Binary Program



Variables $x \in \{0, 1\}^{F \times G}$ with

$$x_{i\alpha} = \begin{cases} 1, & \text{if flight } i \text{ takes gate } \alpha, \\ 0, & \text{otherwise} \end{cases}$$

Minimizing the total transfer time with objective function

$$\begin{aligned} O(x) &= O_{\text{arr}}(x) + O_{\text{dep}}(x) + O_{\text{transfer}}(x) \\ &= \underbrace{\sum_{i\alpha} n_i^{\text{arr}} t_{\alpha}^{\text{arr}} x_{i\alpha} + \sum_{i\alpha} n_i^{\text{dep}} t_{\alpha}^{\text{dep}} x_{i\alpha}}_{\text{linear}} + \underbrace{\sum_{ij\alpha\beta} N_{ij} T_{\alpha\beta} x_{i\alpha} x_{j\beta}}_{\text{quadratic}} \end{aligned}$$



FGA BP Constraints

1. One gate per flight



FGA BP Constraints

1. One gate per flight

$$\sum_{\alpha} x_{i\alpha} = 1 \quad \forall i \in F$$



FGA BP Constraints

1. One gate per flight

$$\sum_{\alpha} x_{i\alpha} = 1 \quad \forall i \in F$$

2. Different gates if standing times of two flights overlap
forbidden pairs



FGA BP Constraints

1. One gate per flight

$$\sum_{\alpha} x_{i\alpha} = 1 \quad \forall i \in F$$

2. Different gates if standing times of two flights overlap
forbidden pairs

$$P = \left\{ (i, j) \in F^2 : t_i^{\text{in}} < t_j^{\text{in}} < t_i^{\text{out}} + t^{\text{buff}} \right\}$$



FGA BP Constraints

1. One gate per flight

$$\sum_{\alpha} x_{i\alpha} = 1 \quad \forall i \in F$$

2. Different gates if standing times of two flights overlap
forbidden pairs

$$P = \left\{ (i, j) \in F^2 : t_i^{\text{in}} < t_j^{\text{in}} < t_i^{\text{out}} + t^{\text{buff}} \right\}$$

$$x_{i\alpha} + x_{j\alpha} \leq 1 \quad \forall (i, j) \in P \quad \forall \alpha \in G$$



FGA BP Constraints

1. One gate per flight

$$\sum_{\alpha} x_{i\alpha} = 1 \quad \forall i \in F$$

2. Different gates if standing times of two flights overlap
forbidden pairs

$$P = \left\{ (i, j) \in F^2 : t_i^{\text{in}} < t_j^{\text{in}} < t_i^{\text{out}} + t^{\text{buff}} \right\}$$

$$x_{i\alpha} + x_{j\alpha} \leq 1 \quad \forall (i, j) \in P \quad \forall \alpha \in G$$

$$\Updownarrow$$

$$x_{i\alpha} \cdot x_{j\alpha} = 0$$



QUBO Formulation

$$Q(x) = O(x) + \lambda_{\text{one}} C_{\text{one}}(x) + \lambda_{\text{not}} C_{\text{not}}(x)$$



QUBO Formulation

$$Q(x) = O(x) + \lambda_{\text{one}} C_{\text{one}}(x) + \lambda_{\text{not}} C_{\text{not}}(x)$$

with penalty terms



QUBO Formulation

$$Q(x) = O(x) + \lambda_{\text{one}} C_{\text{one}}(x) + \lambda_{\text{not}} C_{\text{not}}(x)$$

with penalty terms

$$C_{\text{one}}(x) = \sum_i \left(\sum_{\alpha} x_{i\alpha} - 1 \right)^2$$



QUBO Formulation

$$Q(x) = O(x) + \lambda_{\text{one}} C_{\text{one}}(x) + \lambda_{\text{not}} C_{\text{not}}(x)$$

with penalty terms

$$C_{\text{one}}(x) = \sum_i \left(\sum_{\alpha} x_{i\alpha} - 1 \right)^2$$

$$C_{\text{not}}(x) = \sum_{\alpha} \sum_{(i,j) \in P} x_{i\alpha} x_{j\alpha}$$



QUBO Formulation

$$Q(x) = O(x) + \lambda_{\text{one}} C_{\text{one}}(x) + \lambda_{\text{not}} C_{\text{not}}(x)$$

with penalty terms

$$C_{\text{one}}(x) = \sum_i \left(\sum_{\alpha} x_{i\alpha} - 1 \right)^2$$

$$C_{\text{not}}(x) = \sum_{\alpha} \sum_{(i,j) \in P} x_{i\alpha} x_{j\alpha}$$

where

$$C_{\text{one/not}} \begin{cases} > 0, & \text{if constraint is violated} \\ = 0, & \text{if constraint is fulfilled} \end{cases}$$



QUBO Formulation

$$Q(x) = O(x) + \lambda_{\text{one}} C_{\text{one}}(x) + \lambda_{\text{not}} C_{\text{not}}(x)$$

with penalty terms

$$C_{\text{one}}(x) = \sum_i \left(\sum_{\alpha} x_{i\alpha} - 1 \right)^2$$

$$C_{\text{not}}(x) = \sum_{\alpha} \sum_{(i,j) \in P} x_{i\alpha} x_{j\alpha}$$

where

$$C_{\text{one/not}} \begin{cases} > 0, & \text{if constraint is violated} \\ = 0, & \text{if constraint is fulfilled} \end{cases}$$

Contribution of objective function should not exceed penalty!



Choice of Penalty Weights

Need to ensure that a solution always fulfills constraints, hence

$$\Delta C > \Delta O$$



Choice of Penalty Weights

Need to ensure that a solution always fulfills constraints, hence

$$\Delta C > \Delta O$$

Comparing coefficients in worst cases for



Choice of Penalty Weights

Need to ensure that a solution always fulfills constraints, hence

$$\Delta C > \Delta O$$

Comparing coefficients in worst cases for

- not assigning a flight to any gate

$$\lambda_{\text{one}} > \max_{i, \alpha} \left(n_i^{\text{dep}} t_{\alpha}^{\text{dep}} + n_i^{\text{arr}} t_{\alpha}^{\text{arr}} + \max_{\beta} T_{\alpha\beta} \sum_j N_{ij} \right)$$



Choice of Penalty Weights

Need to ensure that a solution always fulfills constraints, hence

$$\Delta C > \Delta O$$

Comparing coefficients in worst cases for

- not assigning a flight to any gate

$$\lambda_{\text{one}} > \max_{i,\alpha} \left(n_i^{\text{dep}} t_{\alpha}^{\text{dep}} + n_i^{\text{arr}} t_{\alpha}^{\text{arr}} + \max_{\beta} T_{\alpha\beta} \sum_j N_{ij} \right)$$

- assigning a pair of forbidden flights to the same gate

$$\begin{aligned} \lambda_{\text{not}} > \max_{i,\alpha,\gamma} \left((n_i^{\text{dep}} t_{\alpha}^{\text{dep}} - n_i^{\text{dep}} t_{\gamma}^{\text{dep}}) + (n_i^{\text{arr}} t_{\alpha}^{\text{arr}} - n_i^{\text{arr}} t_{\gamma}^{\text{arr}}) \right. \\ \left. + \max_{\beta} (T_{\alpha\beta} - T_{\gamma\beta}) \sum_j N_{ij} \right) \end{aligned}$$



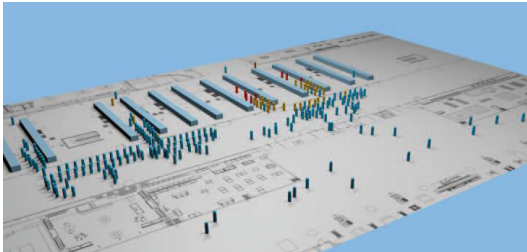
Airport Data

- Flight schedule for one day from a mid-sized European airport



Airport Data

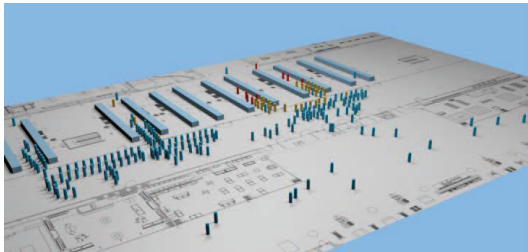
- Flight schedule for one day from a mid-sized European airport
- Passenger flow from agent-based simulation of Martin Jung



Simulating a multi-airport region to foster individual door-to-door travel, M. Jung et al.

Airport Data

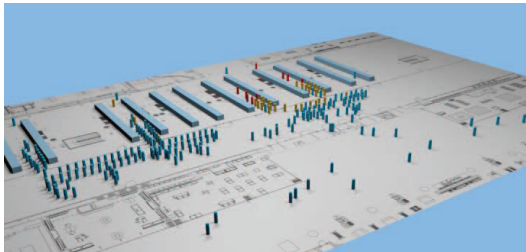
- Flight schedule for one day from a mid-sized European airport
- Passenger flow from agent-based simulation of Martin Jung
- Extracted total instance: 293 flights and 97 gates



Simulating a multi-airport region to foster individual door-to-door travel, M. Jung et al.

Airport Data

- Flight schedule for one day from a mid-sized European airport
 - Passenger flow from agent-based simulation of Martin Jung
 - Extracted total instance: 293 flights and 97 gates
- ⇒ Over 28000 binary variables with about 400 Mio. couplings

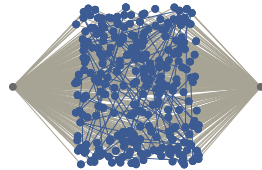


Simulating a multi-airport region to foster individual door-to-door travel, M. Jung et al.

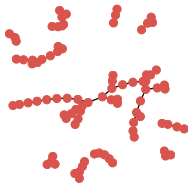
Instance Preprocessing

- Removing corrupted data

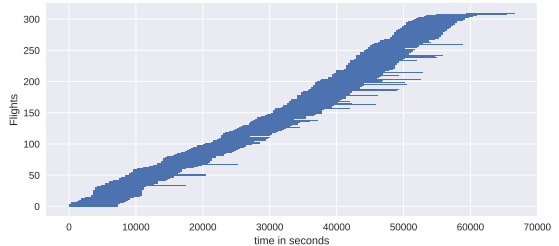
Total problem



Transfer Passengers



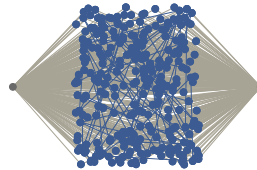
Time at the gates for all flights



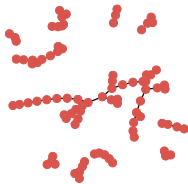
Instance Preprocessing

- Removing corrupted data
- Splitting too long on-block times

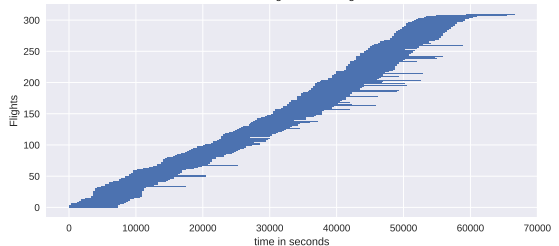
Total problem



Transfer Passengers



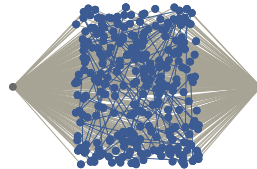
Time at the gates for all flights



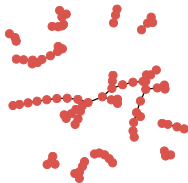
Instance Preprocessing

- Removing corrupted data
- Splitting too long on-block times
- Reducing to flights with transfers only

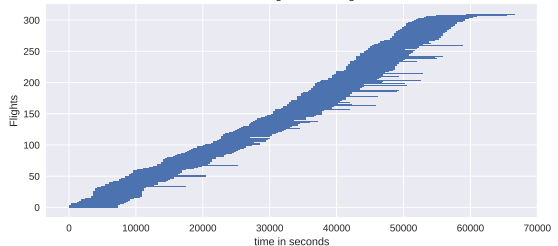
Total problem



Transfer Passengers



Time at the gates for all flights

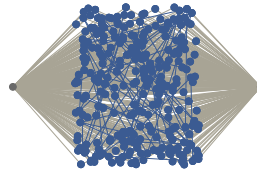


Instance Preprocessing

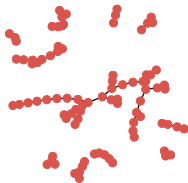
- Removing corrupted data
- Splitting too long on-block times
- Reducing to flights with transfers only

⇒ 89 remaining flights with 80 transfers

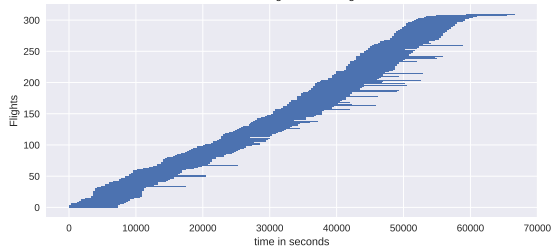
Total problem



Transfer Passengers



Time at the gates for all flights



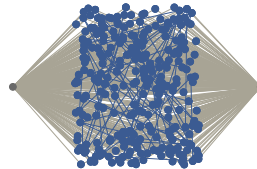
Instance Preprocessing

- Removing corrupted data
- Splitting too long on-block times
- Reducing to flights with transfers only

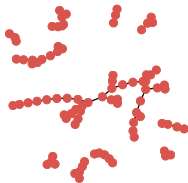
⇒ 89 remaining flights with 80 transfers

⇒ more than 35 gates

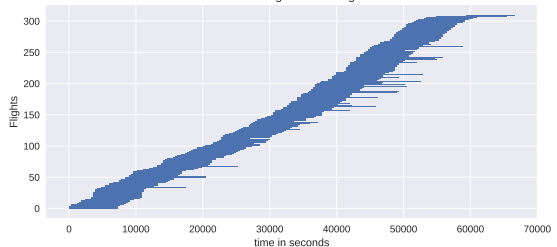
Total problem



Transfer Passengers



Time at the gates for all flights

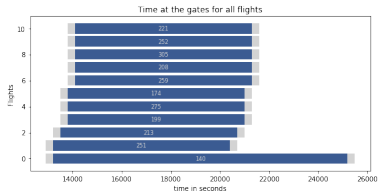
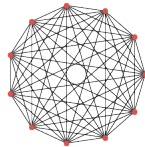


Instance Reduction

- Slicing by time intervals

Transfer Passengers

Forbidden Flight Pairs

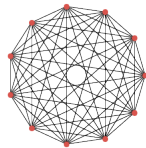


Instance Reduction

- Slicing by time intervals
- Connected components of transfer passenger graph

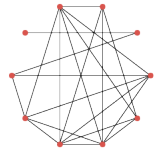
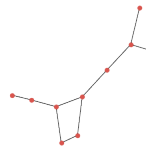
Transfer Passengers

Forbidden Flight Pairs

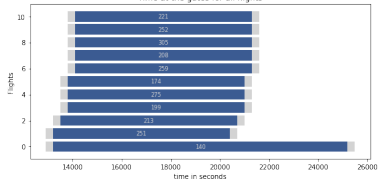


Transfer Passengers

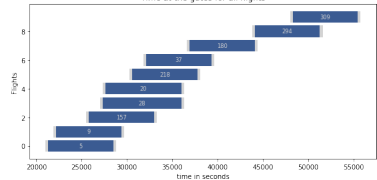
Forbidden Flight Pairs



Time at the gates for all flights



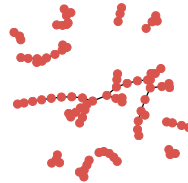
Time at the gates for all flights



Instance Reduction

- Use connected components of transfer passenger graph to find small hard instances

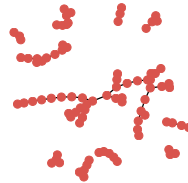
Transfer Passengers



Instance Reduction

- Use connected components of transfer passenger graph to find small hard instances
- Choose number of gates close number of maximal clique

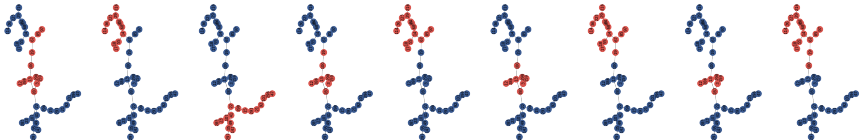
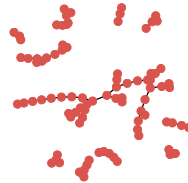
Transfer Passengers



Instance Reduction

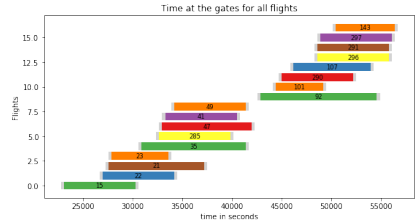
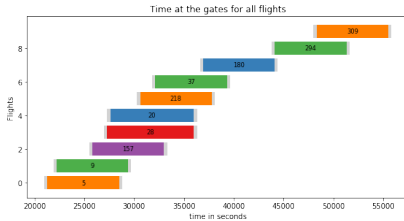
- Use connected components of transfer passenger graph to find small hard instances
- Choose number of gates close number of maximal clique
- Split largest connected component of the transfer passenger graph

Transfer Passengers



Classical Solution

- Getting exact solutions using classical solver SCIP



Embedding

- Standard D-Wave API



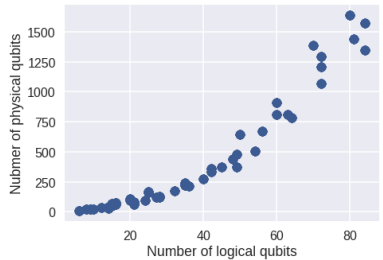
Embedding

- Standard D-Wave API
- 5 embeddings for each instance



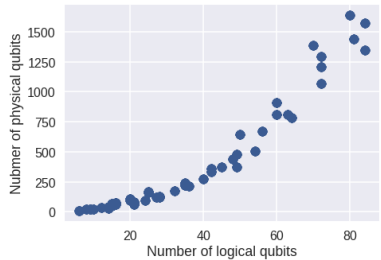
Embedding

- Standard D-Wave API
- 5 embeddings for each instance
- Quadratic overhead



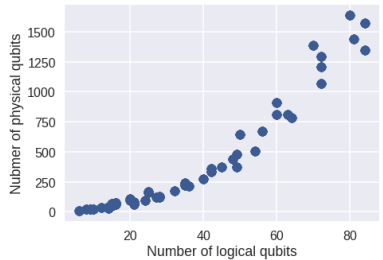
Embedding

- Standard D-Wave API
- 5 embeddings for each instance
- Quadratic overhead
- Maximum 90 logical qubits



Embedding

- Standard D-Wave API
- 5 embeddings for each instance
- Quadratic overhead
- Maximum 90 logical qubits
($\#Variables = \#Flights \cdot \#Gates$)



Annealing Results

- 10000 runs with different parameter settings



Annealing Results

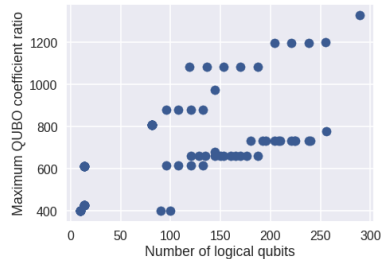
- 10000 runs with different parameter settings
- Even for the small instances:



Annealing Results

- 10000 runs with different parameter settings
- Even for the small instances:

Ratio of maximal to minimal coefficient too large!

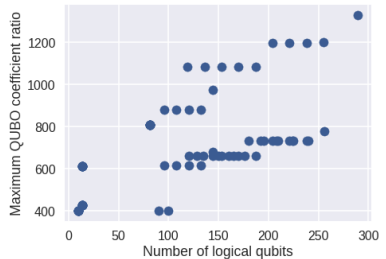


Annealing Results

- 10000 runs with different parameter settings
- Even for the small instances:

Ratio of maximal to minimal coefficient too large!

⇒ Success probability close to 0



Annealing Results

- Running structurally equivalent but random instances with much smaller ratios



Annealing Results

- Running structurally equivalent but random instances with much smaller ratios
- 10000 runs with different parameter settings



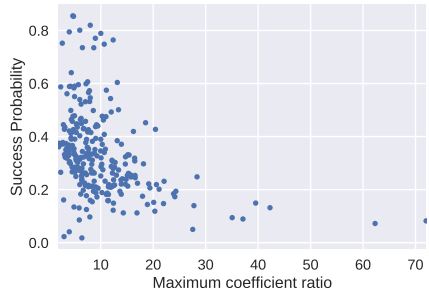
Annealing Results

- Running structurally equivalent but random instances with much smaller ratios
- 10000 runs with different parameter settings
- Differentiated results



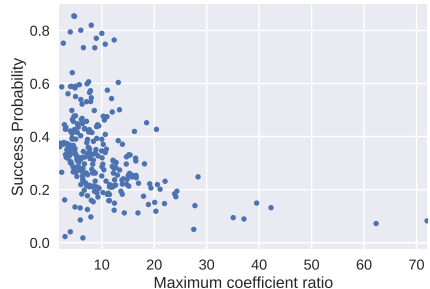
Annealing Results

- Running structurally equivalent but random instances with much smaller ratios
- 10000 runs with different parameter settings
- Differentiated results
 - ratio needs to be < 30



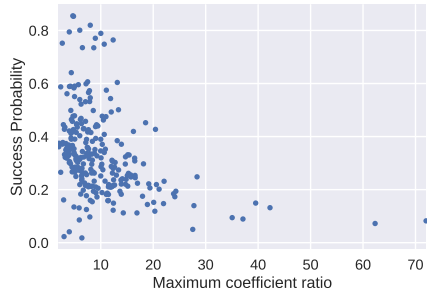
Annealing Results

- Running structurally equivalent but random instances with much smaller ratios
- 10000 runs with different parameter settings
- Differentiated results
 - ratio needs to be < 30
 - wide distribution



Annealing Results

- Running structurally equivalent but random instances with much smaller ratios
- 10000 runs with different parameter settings
- Differentiated results
 - ratio needs to be < 30
 - wide distribution
 - due to other parameters like intra chain coupling of embedding



Outlook

- Adjust embedding/annealing parameters for more clear results



Outlook

- Adjust embedding/annealing parameters for more clear results
- Investigate new D-Wave features



Outlook

- Adjust embedding/annealing parameters for more clear results
- Investigate new D-Wave features
- Using hybrid approach to incorporate classical results



Outlook

- Adjust embedding/annealing parameters for more clear results
- Investigate new D-Wave features
- Using hybrid approach to incorporate classical results
- Using QAOA algorithm (related gate based approach)



Thank You

